CLAIMS:

1. An application programming interface for a programmable graphics processor, comprising:

one or more program instructions to configure a fragment processor within the programmable graphics processor to detect a position conflict for a position and prevent a subsequent access of the position until the position conflict is resolved.

2. The application programming interface of claim 1, wherein a program instruction receives as input a source location and a destination location.

3. The application programming interface of claim 2, wherein the source location includes a buffer identifier corresponding to one of several buffers.

4. The application programming interface of claim 2, wherein the destination location includes a buffer identifier corresponding to one of several buffers.

5. The application programming interface of claim 2, wherein the destination location contains fragment data including at least one of depth, color, and stencil.

6. A method of processing fragment program instructions comprising:

receiving a pixel load instruction including a source address corresponding to a location within the buffer;

detecting a write to the source address is pending; and

waiting to read data stored in the location corresponding to the source address until the write to the source address is complete.

7. The method of claim 6, further comprising storing the data read from the location corresponding to the source address in a register specified by the pixel load instruction.

8. The method of claim 6, further comprising:

receiving additional fragment program instructions after the receiving of the pixel load instruction; and

waiting to execute the additional fragment program instructions until the write to the source address is complete.

9. The method of claim 6, further comprising:

receiving additional fragment program instructions after the receiving of the pixel load instruction; and

executing at least one of the additional fragment program instructions before the write to the source address is complete.

10. The method of claim 6, further comprising:

executing at least one subsequent fragment program instruction while waiting to read the data stored in the location corresponding to the source address.

11. A fragment program for processing fragment data in a fragment processing pipeline, comprising:

a fragment program instruction to write a destination location in a buffer; and

a fragment program instruction to read the destination location in the buffer, without an intervening instruction to flush the fragment processing pipeline.

12. The fragment program of claim 11, wherein the destination location includes a buffer identifier corresponding to one of several buffers.

13. The fragment program of claim 11 comprising fragment program instructions to configure the fragment processing pipeline to perform depth buffering prior to shading.

14. The fragment program of claim 11, comprising fragment program instructions to configure the fragment processing pipeline to perform depth peeling.

15. The fragment program of claim 11, comprising:

fragment program instructions to configure the fragment processing pipeline to perform raster operations.

16. The fragment program of claim 11, wherein the raster operations are performed using fragment data represented in a floating-point data format.

17. A computer program product having a computer readable medium having computer program instructions recorded thereon, the computer program product comprising:

a fragment program for execution by a fragment processing pipeline, the fragment program including:

a fragment program instruction to write a position in a buffer; and

a fragment program instruction to read the position in the buffer, without an intervening instruction to flush the fragment processing pipeline.

18. The computer program product of claim 17, wherein the fragment program includes fragment program instructions to configure the fragment processing pipeline to perform depth buffering prior to shading.

19. The computer program product of claim 17, wherein the fragment program includes fragment program instructions to configure the fragment processing pipeline to perform depth peeling.

20. The computer program product of claim 17, wherein the fragment program includes fragment program instructions to configure the fragment processing pipeline to perform dual depth shadow mapping.

21. The computer program product of claim 17, wherein the fragment program includes fragment program instructions to configure the fragment processing pipeline to perform raster operations.

22. The computer program product of claim 21, wherein the raster operations are performed using fragment data represented in a floating-point data format.